REMARKS

Claims 1, 2, 6, 8, and 16 were pending in the present application. Applicant has amended claims 1 and 6. Claims 1, 2, 6, 8, and 16 remain pending.

§ 101 Rejections

The Examiner rejected claims 1, 2, 6, and 8 under 35 U.S.C. § 101 because the claimed invention lacks patentable utility. Specifically, the Examiner found that the "claimed data structure is not drawn to a practical application which has a useful result because the data structure is not interrelated with software components and hardware components of a computer system to produce an output and thus no result is obtained and thus the consideration of usefulness is moot." August 26, 2006 Final Office Action, p. 2. Applicant respectfully traverses.

The invention of the present application improves the efficiency of prior art tree structures for storing hierarchical data. As well known to one skilled in the art, tree structures are used extensively used in the field of computer science to store hierarchical data, manipulate hierarchical data, and search through hierarchical data. "As Fig. 1 illustrates, each node must have N pointers to its child node. If N is a large number, then tree structure 10 consume a large amount of memory just to store the pointers. Thus, what is needed is a more efficient tree structure." Present application, paragraph [0003]. Accordingly, the present application seeks to provide a more efficient tree structure for storing hierarchical data.

Furthermore, MPEP §2106 provides:

> (a) Functional Descriptive Material: "Data Structures" Representing Descriptive Material Per Se or Computer Programs Representing Computer Listings Per Se
>
> Data structures not claimed as embodied in computer-readable media are descriptive material per se and are not statutory because they are not capable of causing functional change in the computer. See, e.g., Warmerdam, 33 F.3d at 1361, 31 USPQ2d at 1760 (claim to a data structure per se held nonstatutory). Such claimed data structures do not define any structural and functional interrelationships between the data structure and other claimed aspects of the invention which permit the data structure's functionality to be realized. In contrast, a claimed computer-readable medium encoded with a data structure defines structural and functional interrelationships between the data structure and the computer software and hardware components which permit the data structure's functionality to be realized, and is thus statutory.

MPEP § 2106, 2100-13 (emphasis added). As the Examiner can see, a computer readable medium encoded with a data structure is statutory since the data structure inherently defines structural and functional interrelationships between the data structure and the computer software and hardware components to permit the data structure's functionality to be realized. This is certainly true for claims 1, 2, 6, and 8 as their recited pointers provide structural and functional interrelationships between the recited nodes.

For the above reasons, Applicant respectfully requests the Examiner to withdraw the § 101 rejections of claims 1, 2, 6, and 8.

§ 112 Rejections

The Examiner rejected claim 1 under 35 U.S.C. § 112, first paragraph, for failing to comply with the written description requirement. Specifically, the Examiner found that the specification does not support the claim language of "last traversed in data structure" and the specification does not describe what is meant by "always pointing to a child node that was last traversed" since Fig. 2 shows two pCursor pointers. August 26, 2006 Final Office Action, p. 3. Applicant respectfully traverses and submits that the specification supports the claim languages as follows.

As can be seen in Fig. 2, a parent node (e.g., node 52) in a data structure 50 has pointers pHead, pTail, and pCursor to child nodes in the next hierarchical level (e.g., the 1st level with nodes 54-0 to 54-N) of structure 50. If pCursor is initially null, one can use pointers pHead and pTail of parent node 52 to reach the child nodes in the 1st hierarchical level and then update pCursor of parent node 52 with the location of the last traversed child node in the 1st hierarchical level. The last traversed child node in the 1st hierarchical level is the child node that was last walked before going into the 2nd hierarchical level or when the search ends at the 1st hierarchical level (e.g., when there is no more hierarchical levels). As can be seen, pCursor of parent node 52 always points to the last child node that was traversed from parent node 52. The same structure is repeated for other nodes and therefore each node may have pointers pHead, pTail, and pCursor. These steps are illustrated in steps 86, 88, 90, and 92 in the flowchart of Fig. 4 and described in paragraphs [0017] to [0021]. To further clarify the claim language, Applicant has also amended claim 1 to recite "a child node that was last traversed from the parent node in data access." Amended claim 1 (emphasis added). Thus, the specification fully supports the claim language of amended claim 1.

The Examiner rejected claim 6 under 35 U.S.C. § 112, second paragraph, for failing to particularly point out and distinctly claim the subject matter of the invention. Specifically, the Examiner found the claim language of "retrieving another data from the data store" to be unclear. Applicant notes that the claim language in question appears in claim 16 and actually recites "retrieving another data from the data structure." As described in the specification, a data structure is often requested (e.g., read or written) in order.

> In most applications, nodes are requested in order (forward or backward). For example, nodes 54-0 to 54-N are requested in order. Thus, the quickest path to the requested node results from following pointer pCursor of root node 52 to the last requested node, and then following pointer pNext or pPreview to the currently requested node. In another example, nodes 58-0 to 58-N are requested in order. Thus, the quickest path to the requested node most often results in following pointer pCursor of root node 52 to node 54-2, then following pointer pCursor of node 54-2 to the last requested node, and then following pointer pNext or pPreview to the currently requested node. Thus, the quickest path most often results from following pointers pCursor to the level of the currently requested node, and then following pointer pNext or pPreview to the currently requested node.

Specification, paragraph [0018]. Thus, the claim language in claim 6 describes a first request for one data in the data structure and the claim language in claim 16 describes a second request for another data in the same data structure. Applicant submits that one skilled in the art would clearly understand the claim language of "retrieving another data from the data structure" as a subsequent request for another data in the same data structure.

Accordingly, Applicant respectfully requests the Examiner to withdraw the § 112 rejections to claims 1 and 16.

<u>§ 103 Rejections</u>

The Examiner rejected claims 1, 2, 6, 8, and 16 under 35 U.S.C. § 103(a) as being unpatentable over U.S. Patent No. 6,185,569 ("East et al.") in view of U.S. Patent App. Pub. No. 2001/0014097 ("Beck et al.").

<u>Claim 1</u>

Addressing claim 1, the Examiner stated:

> East discloses the elements of the claimed invention as noted above but does not disclose a first pointer always pointing to a child node that was last traversed in

> data access. Beck discloses a first pointer always pointing to a child node that was last traversed in data access [paragraph 46]. It would have been obvious to one of ordinary skill in the art at the time the invention was made to modify East to include a first pointer always pointing to a child node that was last traversed in data access as taught by Beck for the purpose of complying with a round-robin access routine [paragraph 46].

August 26, 2006 Final Office Action, pp. 4 and 5 (emphasis added). Applicant respectfully traverses.

"In order to rely on a reference as a basis for rejection of an applicant's invention, the reference must either be in the field of the applicant's endeavor or, if not, then be reasonably pertinent to the particular problem with which the inventor was concerned." In re Oetiker, 977 F.2d 1443, 1446 (Fed. Cir. 1992). "While Patent Office classification of references and the cross-references in the official search notes are some evidence of "nonanalogy" or "analogy" respectively, the court has found 'the similarities and differences in structure and function of the inventions to carry far greater weight.'" MPEP § 2141.01(a).

Beck et al. is not in the same field as Applicant's endeavor. The present application is related to tree structures for storing data. Present application, paragraph [0001]. Unlike the present application, Beck et al. is related to the clustering of processor nodes (e.g., computers) connected by a subnet to communicate with other processor nodes. Beck et al., paragraphs [0001].

Beck et al. is not related to the problem which the Applicant is concerned. The present application is concerned about improving the efficiency of tree structures for storing data by introducing a new type of pointers that takes advantage of the fact that data requests normally occur in order. Present application, paragraphs [0003] and [0018]. On the other hand, Beck et al. is concerned about presenting a cluster of processor nodes as a single processor node without incurring detrimental overhead. Beck et al., paragraphs [0007] to [0010].

Beck et al. is different in structure than Applicant's invention. The present application discloses a tree structure for storing data. Present application, Fig. 2. On the other hand, Beck et al. discloses a cluster of processor nodes (e.g., computers) connected to each other through a subnet and then to a client processor through a network. Beck et al., Fig. 2.

Beck et al. is different in function than Applicant's invention. The present application provides a tree structure and a related method for storing data. Present application, Figs. 2 and 4. On the other hand, Beck et al. provides a round robin algorithm for selecting a processor node in a cluster of processor nodes to handle a new connection request from a client processor. Beck et al., paragraph

[0046]. More specifically, the round robin algorithm considers the last processor node marked by a software pointer before other candidate processor nodes. Beck et al., paragraph [0046].

For all of the above reasons, Beck et al. is not analogous art with the claimed invention and therefore cannot be combined with East et al. in a § 103(a) rejection.

Furthermore, the Examiner has not provided a proper motivation to combine East et al. and Beck et al. East et al., like the present application, is not about clustering processor nodes or handling connection requests to a cluster of processor nodes. Thus, there cannot be any reason why East et al. would use the software pointer of the round robin algorithm of Beck et al. in the data structures of East et al.

Claims 2, 6, 8, and 16

Claim 2 depends from amended claim 1 and is patentable over East et al. and Beck et al. for at least the same reasons as amended claim 1.

Claim 6 recites similar language as amended claim 1. Accordingly, amended claim 6 is patentable over East et al. and Beck et al. for at least the same reasons as amended claim 1. Claims 8 and 16 depend from amended claim 6 and are patentable over East et al. and Beck et al. for at least the same reasons as amended claim 6.

Summary

In summary, claims 1, 2, 6, 8, and 16 were pending in the above-identified application when last examined. Applicant has amended claims 1 and 6. For the above reasons, Applicant respectfully requests the allowance of claims 1, 2, 6, 8, and 16. Should the Examiner have any questions, please call the undersigned at (408) 382-0480.

Respectfully submitted,

/David C Hsia/

David C. Hsia
Attorney for Applicant(s)
Reg. No. 46,235
Patent Law Group LLP
2635 North First St., Ste. 223
San Jose, California 95134
408-382-0480x206